# 1, 2, 3, кодирај ! – 5-6 разред основне школе - Етапе 2.8: Учинимо игру интересантнијом

| | |
|---|---|
| Резиме | Ученици комплетирају видео игру додавањем неколико елемената који је чине занимљивијом. На пример, бројањем уназад, торнадо који се креће насумично и све брже и брже, итд. Користе концепте које су упознали на претходним часовима: тест, петља, променљива, догађај. |
| Појмови | « Машине» :<br><br>• Машине које се налазе у нашем окружењу само извршавају наредбе (инструкције).<br>• Комбинацијом више елементарних инструкција ми им омогућујемо да изврше комплексне задатке.<br><br>« Алгоритми »<br><br>• Алгоритам омогућује да решимо неки проблем.<br>• Петља омогућује да се иста активност понови више пута<br>• Неке петље се никад не заустављају па носе назив « бесконачне ».<br>• Неке петље се понављају одређен број пута и називају се « итеративне ».<br>• Неке петље, назване « условним », се понаваљају до испуњења неког услова.<br><br>« Језик » :<br><br>• За давање инструкција некој машини користимо неки програмски језик, разумљив и машини и човеку.<br>• *Скрач* радно окружење графичког програмирања користи једноставан језик.<br>• Неки програм је представљен алгоритмом у неком програмском језику.<br>• Неке инструкције се извршавају само кад се деси неки догађај па говоримо о програмирању вођеног догађајима.<br>• Неке инструкције се извршавају једна за другом па кажемо да је реч о секвенцијалном програмирању.<br>• Извршење неког програма је репродуктибилно (ако се не мењају ни инструкције ни подаци којим манипулишемо, онда програм даје увек исти резултат) |

| Материјал | За сваки пар ученика<br><br>• Компјутер са *Скрачом* и меморисаним програмом са претходног часа |
|---|---|

## Полазна ситуација

Ученици су сигурно запазили да могу да играју ову игру али и да она није тако интересантна јер не садржи неке веће потешкоће. Ако пазимо на препреке та игра може да се настави у недоглед.

Одељење заједнички разматра како да заврши игру уводећи неку потешкоћу. За тако нешто постоји више могућности:

- Опција 1: додајемо бројање уназад. Када је расположиво време истекло, игра се зауставља. Перформансе играча се процењују бројем преосталих « живота » и бројем прикупљених ресурса (његов « скор »).
- Опција 2: додавањем неког новог елемента који ће игру учинити тежом и довести до њеног заустављања у неком тренутку. На пример, можемо увести нову клопку (торнадо, неки ехо у метеоролошким проблемима поменут на часу 1.3 и часу 1.4, нешто у вези бинарног кода) креће све брже и брже а ми нисмо у стању да предвидимо његов правац и у једном тренутку игра се зауставља.

**Педагошке напомене:**

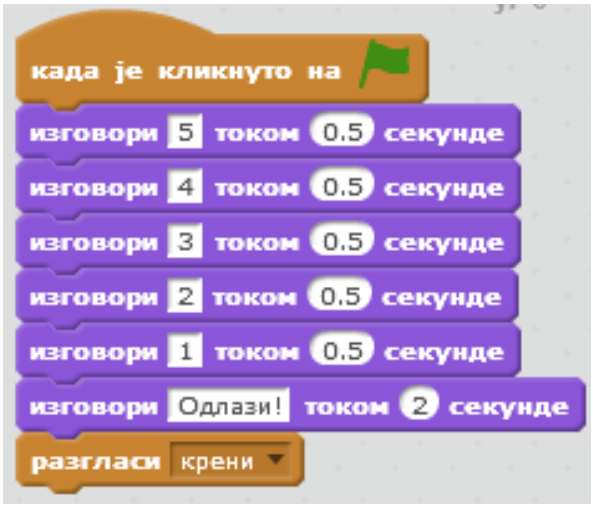
- Велика је вероватноћа да ученици замисле неки други начин завршетка игре. Наставнику саветујемо да покрене дебату у одељењу с циљем да се одаберу опције која ће се следити (није сигурно да ће све групе одабрати исту опцију!).
- Наставник ће пратити рад ученика водећи рачуна о изводљивости предложених идеја. Ако је нека идеја атрактивна али не и релано изводљива, боље је определити се за неку другу коју ће с великм вероватноћом реализовати!

У наставку ћемо описати 2 поменуте опције, као и неколико других које не зачињавају игру, али је чине привлачнијом или успешнијом при избегавању неких багова. Дољи задаци су различите тежине и независни су једни од других.

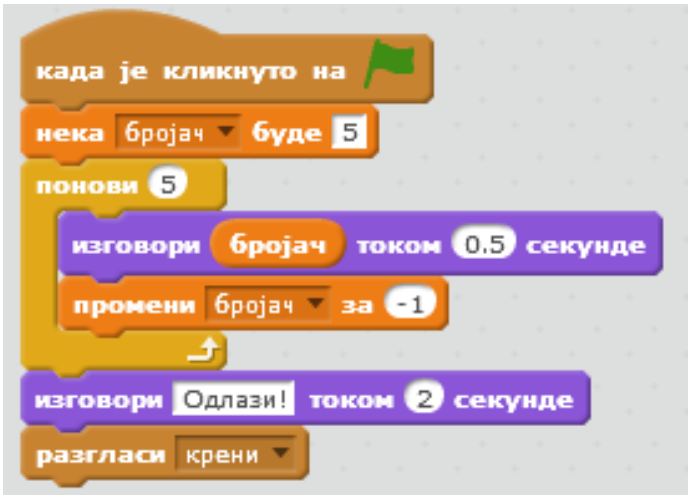
**Наш савет: урадите бар задатак 2 или задатак 3** како би игра заинтересовала ученике.

## Задатак 1 : покрените игру реализујући бројање уназад (15 минута)

Можемо одлучити да се игра покрене само после завршетка бројања уназад *5, 4, 3, 2, 1, крећи !* Ово можемо урадити врло једноставно на следећи начин:




… или знатно суптилније уз коршћење нове променљиве и петље.



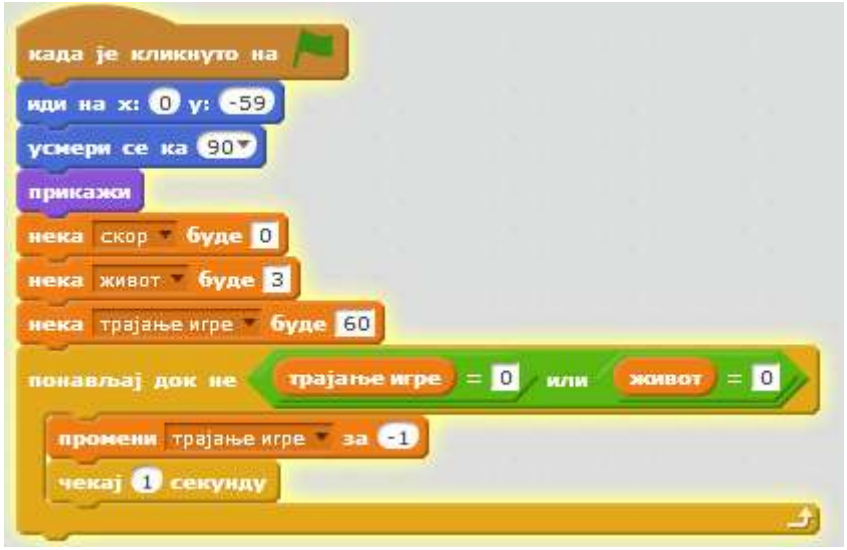

Зависно од нивоа ученика, можемо се задовољити давањем једноставног начина или им можемо предложити да користе једну променљиву и једну петљу. Код комплекснијих варијанти можемо дати све елементе програма али хаотично распоређене и међусобно неповезане, а ученци би требало да их поставе тако да програм ради и даје резултате.

## Задатак 2 : ограничите време трајања игре (15 минута)

**Педагошке напомене:**

- Овај задатак је сличан претходном, али се обавезно користи комплекснија форма (променљива плус петља) јер је незамисливо написати програм код кога се броји уназада, секунд по секунд, током више минута а да се не користи петља.
- Ако су ученици већ направили програм за бројање уназад у (задатак 1), успеће врло лако да реше овај нови задатак.. Ако пак то нису урадили, онда ће им требати мало времена да размисле и можда да добију неку врсту лаке помоћи у вођењу при решавању проблема.
- Овај задатак омогућује да се вратимо на променљиве, тестове, петље, као и на логичке операторе.

Временско ограничавање игре је врло једноставан начин који је чини интересантијом. Довољно је креирати променљиву « трајање игре », даје се почетна вредност која се током времена смањује, увођењем одговарајућег тајминга да би се омогућила контрола брзине одвијања процеса.
Програм се зауставља појавом (поруке « game over »), или када је бројање уназад дошло до 0, или када је број живота достигао вредност 0.
Програм за ровер би морао да буде модификован тако да укључује:




*У овом примеру постављамо да је « трајање игре » 60 и одузима се 1 сваког секунда.*
*Можемо, сасвим сигурно, мењати ову променљиву да би живот био мање или више дуг.*

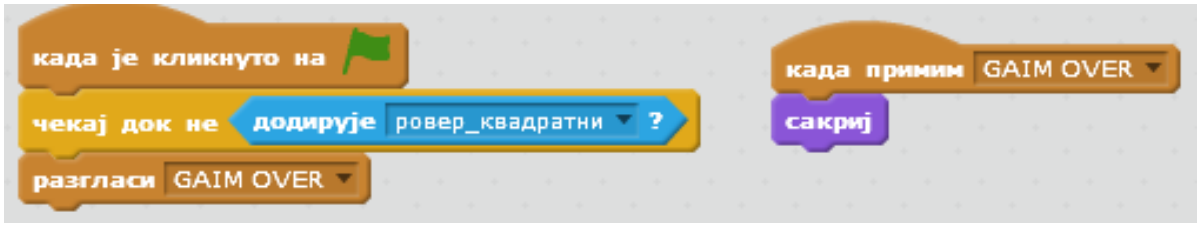
## Задатак 3 : увођење торнада који се насумично креће (15 минута)

Ученици сад знају да креирају лик помоћу слике и одреде његову почетну позицију. На пример, торнадо може да се постави у доњи леви угао екрана (X = -230 и Y = -170).
Затим је потребно да се нађе начи његовог насумичног померања. Ако желимо да његово кретање буде тренутно довољно је да се промене вредности за X и Y…
Међутим, повољније је да видите како се торнадо премешта! Адекватна команда је « клиз **… секунди на x=… и y=…** », расположива у категорији «кретање ».

Пошто желимо да се торнадо умери ка било ком месту на екрану, довољно је да напишемо следећу инструкцију:


Ученици могу кликом на ову инструкцију да виде да се торнадо сваки пут премешта у неком другом правцу. Премештање се увек реализује током 1 секунде, па ако је место доласка торнада близу онда померање изгледа споро, а ако је то мест удаљено онда се померање дешава врло брзо.
Преостаје нам да повежемо ову инструкцију с остатком програма за торнадо (мењамо време на « 2 секунде » јер тако успоравамо померање торнада).


Имајте на уму да ако ровер додирне торнадо онда се игра завршава. Зато у програм за торнадо додајемо следећу инструкцију:

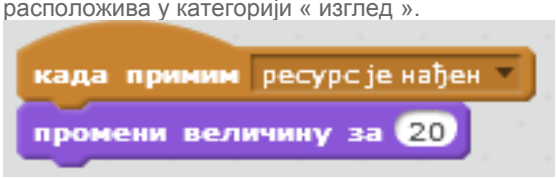



## Задатак 4 : повећај торнадо (15 minutes)

Повећавањем торнада са повећаним бројем прикупљених ресурса игра постаје још интересантнија. Овакав задатак, у овом стадијуму пројекта, не представља неки пробле.

- Потребно је послати поруку, програмима за ресурсе (лед и вегетација), кад год се нађе неки ресурс.


- Повећање торнада се остварује сваки пут када је примљена порука « ресурс је нађен » . За ово користимо команду « **промени за … величину** » која је расположива у категорији « изглед ».



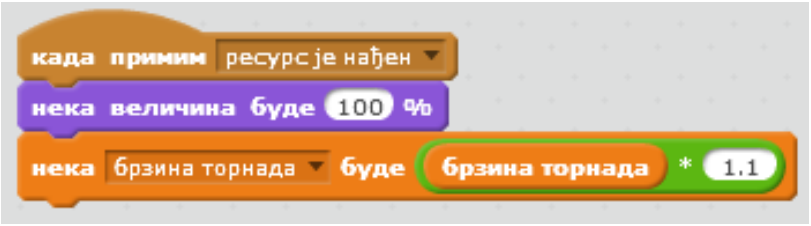
- Пошто сад можемо да модификујемо величину торнада у програму, онда је потребно да мислимо и на почетну вредност торнада при покретању програма. То можемо да урадимо захваљујући команди « **нека величина буде 100%** » која је у категорији « изглед ».

## Задатак 5: убрзај торнадо постепено (20 минута)

Овај задатак је врло тежак и више би одговарао старијим ученицима основне школе и првим разредима средње школе. Ипак постоји могућност да су неки ученици петог или шестог разреда напредовали толико да могу да се носе и са оваквим *изазовима.* Ево чиме би могли да се добровољно забаве!

Торнадо би требало да се убрза сваки пут када се нађе неки ресурс (лед или вегетација). Убрзање торнада се реализује тако што се прво направи променљива «брзина_торнада » (почетна вредност је 1, у истом програму као и за остале променљиве) која се повећава, на пример за 10% за сваки догађај (подсећам да би повећању брзине од 10% могло да одговара и множење с 1,1).

Модификован програм за торнадо би био следећи:




Пажња, потребно је да имате на уму да је вредност « брзина_торнада » у програму који командује његово кретање (инструкција « клизи »). Ово се реализује тако што се « брзина_торнада » укључује у прорачун времена потребног за клизање, на пример овако:

Што је « брзина_торнада » већа, утолико је време за извршење датог премештања краће, јер је торнадо све бржи и бржи што управо и желимо.

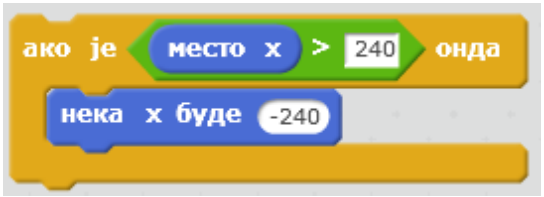
## Задатак 6 : симулација тороидног координатног система (кретање по целој сцени) (20 минута)

Игра је забавнија ако не долази до заустављања ровера при додиру ивица сцене. Потребно је програм направити тако да кад ровер дотакне десну страну сцене, долази до његовог појаве на левој страни сцене и обрнуто. Исто ће се дешавати са горњом и доњом страном сцене.

**Научне напомене**

- Савијањем равни, тако да се споје десна и лева страна ивице, добија се такозвани « цилиндирчни » простор (координатни систем). Ако пак сад додамо још и савијање које омогућује додир горње и доње ивице, онда је реч о « тороидном » простору. Овај простор је сличан крофни или гуми аутомобила.
- Многе видео игре су реализоване у тороидном систем, иако такав систем није сличан оном на реалној планети ( када се нађемо на Северном полу, нисмо истовремено и на Јужном!).

Ученицима преостаје да покушавају насумично или пак да их водимо ако имају потешкоће при решавању проблема. Потребно је да им прво објаснимо алгоритам, јер ако је апсциса X ровера већа од 240 (десна страна екрана), онда он мора да пређе на -240 (лева страна екрана). Затим је потребно да им прикажемо различите блокове који се користе при конструкцији програма, на пример, петља «понављај », контрола « ако … онда », оператор « веће од », вредност променљиве X (плави блок « апсциса X »), и инструкција која омогућује промену ове вредности (плави блок « нека X буде »). Ти блокови се повезују на следећи начин:




**Педагошка напомена**
Могуће је, зависно од облика ликова, ако је потребно, увести малу маргину (на пример, уместо да узмемо 240 као екстремну вредност екрана, можемо да ставимо 235).

Пошто се једном схвати како програмирати прелаз са десне на леву ивицу екрана, онда је веома лако програмирати прелаз од доње на горњу ивицу и од горње на доњу ивицу (променљива Y).
Сада смо, коначно, у могућности да симулирамо ситуацију у ториодном систему:


**Педагошке напомене**

- На екрану се могу дописати коментари на програм, као што је у претходном случају показано. То је веома корисно јер се програм појашњава, а нарочито је корисно за оне који буду користили овај програм!
- Када је достигнут овај ниво, неопходно игнорисати инструкцију « ако си на рубу, окрени се » која је у подпрограмима омогућавала да водимо ровер помоћу стрелица.

## Задатак 7: избегавање додира ресурса и клопки (20 минута)

Овај задатак, као и задатак 5, је намењен ученицима који су знатније напредовали у односу на друге ученике завршног разреда основне и почетног разреда средње школе.
Када су ресурси (лед, вегетација) нађени, поново се појављују на случајном месту на екрану. Постоји могућност да се то место поновног појављивања ресурса

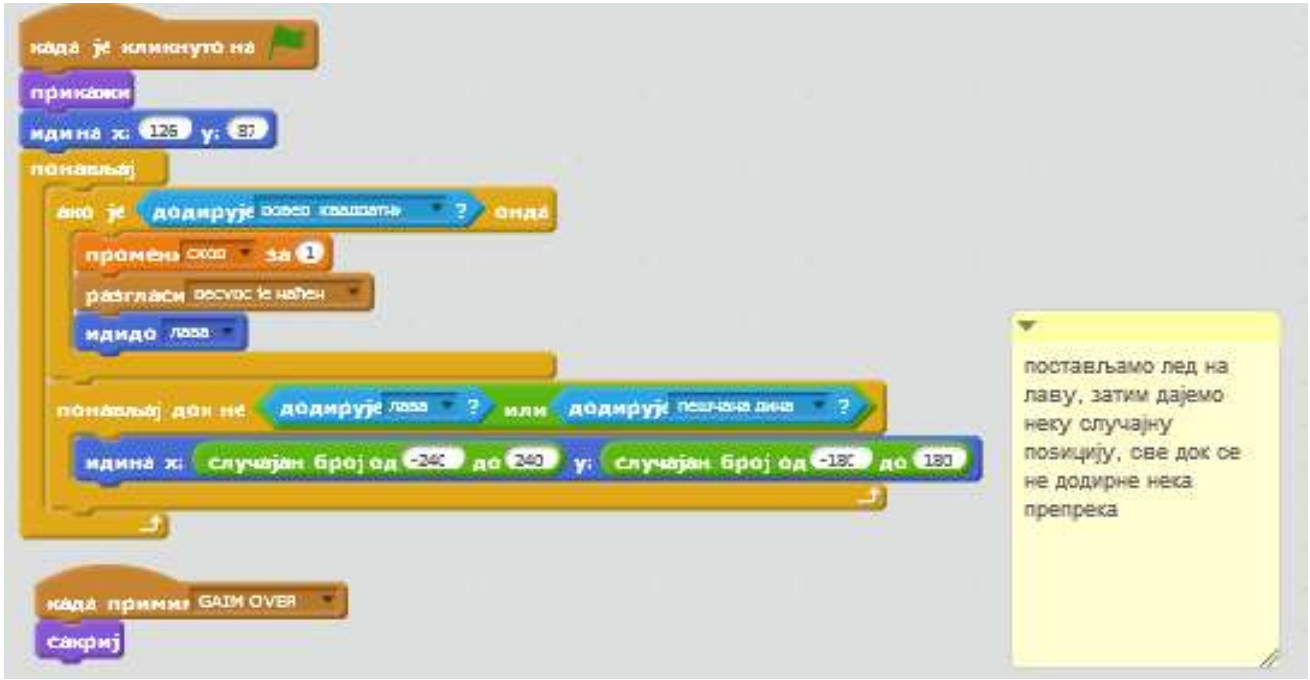
преклопи са местом у ком је нека клопка (дина или лава). Ову ситацију је потребно избећи јер не можемо имати 2 контрадикторна циља, да прикупљамо ресурсе и избегавамо клопке.

Шта да урадимо да се оваква ситуација ипак не појави? Потребно је нову позицију одредити као случајну док ресурс не додирне клопку. Ипак, ресурс у почетном тренутку не додирује клопку, па се петља неће извршити. Вештина се састоји у увођењу неког претходног корака којим би се петља покренула на извршење први пут. Алгоритам тада постаје:

- 1/ Поставите ресурс случајно на било којој координати (или ако желите поставит га на клопку)
- 2/ Затим покрените  петљу: док ресурс није на неком  месту на ком нема клопки, дајте му нову случајну позицију.

Ученици током решавања оваквог проблема морају бити вођени, било давањем горе поменутих досетки, било давањем коначног програма и постављањем питања да ли су спремни на анализу оног што раде и зашто то раде.

Коначни програми за лед и вегетацију су:



## Закључак у писаном облику

Игра је заинтересовала ученике кад могу да је наставе решавајући неке нове изазове

Ученици завршавају листу инструкција које су до сада упознали у *Скрачу*.

Наставник анимира заједнички осврт на он што су ученици научили током реализације овог пројекта, тешкоће с којим су се суочили, евентуалне жеље које су се појавиле (неки ученици су, вероватно, већ почели да праве друге програме у *Скрачу*), итд.

**Педагошка напомене:**

- Овакв тип активности током програмирања је вероватно нов за већину ученика. Саветујемо вам да ученицима, касније током школске године, предложите да направе свој лични пројекат, јер само тако неће заборавити како се користи *Скрач*, а постаће креативнији. Могућности су врло велике, јер могу да направе видео игру (попут пројекта који смо реализовали, што је доста комплексно), а могу и да направе анимирану честитку (за Божић, Нову годину...), или пак интерактивни упитник. Могућности су изузетне!
- Циљ следеће етапе је упознавање неколико нових функционалности *Скрача*, и реализација других идеја.

---

<< Etape III-2.7 Séquence III-2 Etape III-2.9 >>